EXPERIMENTS OF MORPHOGENESIS IN SWARMS OF SIMPLE MOBILE ROBOTS

MARCO MAMEI, MATTEO VASIRANI, FRANCO ZAMBONELLI
Dipartimento di Scienze e Metodi dell'Ingegneria,
Università di Modena e Reggio Emilia

In this paper we focus on the problem of having a multitude of very simple mobile robots self-organize their relative positions so as to obtain a variety of spatial configurations. The problem has a variety of applications in mobile robotics, modular robots, sensor networks, and computational self-assembly. The approach we investigate in this paper attempts at minimizing the local capability of robots and at verifying how and to which extent a variety of global shapes can be obtained by exploiting simple self-organizing algorithms and emergent behaviors. Several experiments are reported showing the effectiveness of the approach.

***Keywords**: Multiagent systems, mobile robots, self-organization, morphogen gradients, emergent behaviors.*

INTRODUCTION

The possibility of manufacturing sub-millimeter-scale computer-based systems with communicating and sensing capabilities (Pister 2003) let us envision the possibility of building sorts of multi-cellular computational organisms, made up of millions of interacting autonomous micro robots, capable of assembling and dynamically re-assembling themselves into a variety of complex shapes for interacting with – and acting in – an environment.

While electronics and communication technologies are advancing, computer scientists and software engineers will be asked to answer two key questions to contribute to the vision: *(i)* How can one achieve reliable patterns of activities in systems made up of a large number of simple components interacting in

unstructured and dynamic networks? *(ii)* How does one translate pre-specified global goals into the local interactions of vast numbers of parts?

In general, the critical task is to identify appropriate (self-)organization principles and programming methodologies for controlling the overall behaviour of such complex systems. Our specific goal in this paper is to study how and to which extent a swarm of very simple mobile robots can be programmed to coordinate their respective movements and create variety of global shapes. Apart from the vision of computational self-assembly (Nagpal 2002; Guo et al. 2004), the problem has also more practical short-term applications: coordinating the movements of navigator-equipped cars (Mamei and Zambonelli 2004b); coordinating the movements of a rescue team provided with PDA (Mamei and Zambonelli 2004a); enforcing self-deployment of sensor networks (Estrin et al. 2002) and of complex robots in a landscape (Bay and Unsal 1994; Fredslund and Mataric 2002).

Biological organisms – achieving coherent, reliable and complex behaviour from the local cooperation of large numbers of identically "programmed" cells – are the most natural source of inspiration for all these kinds of problems. A variety of phenomena hint at powerful underlying mechanisms that can adapt to variation, while maintaining constraints that may be geometric, topological or functional (Lawrence 1992; Wolpert 1998). In particular, as it will be described though the paper, the diffusion of chemicals among cells and the possibility for cells to be driven in their behaviour by the locally sensed gradients of diffused proteins ("morphogen gradients") (Day and Lawrence 2000) seems applicable to the problem of pattern formation in simple mobile robots.

A large number of papers deal with pattern formation in mobile robots (Bay and Unsal 1994; Spears and Gordon 1999; Fredslund and Mataric 2002), and some exploit approaches somewhat similar to the one of morphogen gradients (Shen et al. 2002; Nagpal 2002). The key contribution here is to show how a variety of patterns (from regular to non-regular ones, also involving differentiation in robots) can be achieved – with the use of morphogen gradients and in a scalable and effective way – even in the absence of those capability (e.g., global perception, distance and direction sensing) that are required by most other approaches.

This paper is organized as follows. Section 2 introduces the concept of morphogen gradients. Section 3 details our approach and compares it with related work. Section 4 presents several experiments in pattern formation. Section 5 discusses the performances of our approach. Section 6 concludes.

MORPHOGEN GRADIENTS

Morphogenesis is one of the outstanding problems in biological sciences. It concerns the basic question of how biological shapes are generated starting from an immense number of identical cells. Translating this problem into future nano-

technology scenarios means answering the following question: given a swarm of autonomous mobile micro robots, possibly very simple and with limited capabilities, how can we have them self-organize into any require spatial shape? Although we are still far from a general answer, some recently discovered mechanisms driving biological morphogenesis are turning out to be of use also in computational morphogenesis.

A mechanism common throughout embryo development and recognized as of primary importance towards morphogenesis is the use of *morphogen gradients* to determine positional information and polarity of cell. For instance, in the Drosophilae embryo, cells at one end of the embryo can emit a protein that diffuses along the length of the embryo. The concentration of this protein and its gradient (that is, the morphogen gradient) is used by other undifferentiated cells to determine whether they lie in the head, thorax or abdominal regions. Different morphogen gradients are used to determine the dorsal-ventral axis, wing development, and even leg bristle polarity [Wol98].

Reproducing morphogen gradients in dense network of short-range wirelessly interacting robots is dramatically simple. A "source" robot can create a morphogen gradient by broadcasting a simple message to its local neighborhood (i.e., to all robots within the connection range). The message can be a simple tuple containing a unique name and a value, initially at zero. Neighbor robots re-broadcast the message in their turn after having modified its value, typically incrementing it, and so on, until the morphogen gradient has propagated through the entire population. Each robot stores and forwards only the minimum value it has heard for a particular morphogen name, thus the morphogen gradient typically represents the shortest path from the source.

The above very simple mechanism can be used in powerful ways to influence the local and global activities of robots.

1. *Leader Election***:** Morphogen gradients can be used to elect leaders in the group. Randomly elected leaders could propagate 'leader' gradients through the network inhibiting others to become leaders on their turn (Nagpal 2002).
2. *Selective Propagation***:** Robots can be programmed to selectively choose which morphogen gradients to propagate, also on the basis of the perceived value of other morphogen gradients. Thus, robots can act as barriers/inhibitors to specific morphogen gradients.
3. *Region Selection:* If a leader robot propagates a morphogen gradient, other robots can inhibit its propagation when it reaches a maximum value. Thus, one can create nearly circular regions of controlled size (Figure 1a). If the morphogen gradient value is incremented by one at each step, it provides an estimate of distance from the source: a value of n steps implies a distance nr from the source, where r is the communication range (Nagpal et al 2003).
4. *Coordinate System:* Morphogen gradients can be used to self-organize coordinate systems: robots can recursively evaluate their coordinates by

triangulating the distances – expressed by means of morphogen gradients – from elected beacons (Nagpal et al. 2003).

5. *Patterning of Activities:* When several robots (or all robots) are sources for the same morphogen gradient, and when they can also inhibit the diffusion of some morphogen gradients, complex patterning in robots' activities (Figure 1b) can be created adopting a reaction-diffusion interaction model (Bonabeau 1997).
6. *Communication:* Morphogen gradients can be used as a way to broadcast messages to other robots. By having these messages follow other morphogen gradients previously diffused, several routing mechanism can be enforced (Poor 2001).
7. *Adaptive Morphogens:* Morphogen gradients can update their values to reflect changing conditions (e.g. lose significance if not constantly reinforced, adapt as robots move, appear or disappear) (Mamei and Zambonelli 2004a).

In addition, all the above ways of exploiting morphogenesis can be put to service to another basic mechanism to promote spatial pattern formation in group of mobile robots, which is that of interest in this paper.

8. *Driving Motion:* if a robot can perceive the local slope of any morphogen gradient, it can also move following such slope uphill, downhill or along equipotential lines, as if it were a gravitational field (Mamei and Zambonelli 2004a; Mamei and Zambonelli2004b; Coore 2001).

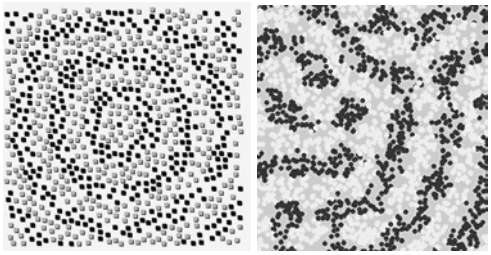

Figure 1. Morphogen gradients: (a) propagating from a source and defining spatial circular spatial regions; (b) propagating from several robots and being inhibited in their propagation, in a sort of reaction-diffusion model.

OUR AND RELATED APPROACHES

In this section, we first describe our model for robots and how robots can exploit morphogen gradients towards pattern formation. Then we present related approaches and possible objections to our model

Our approach

To be "compliant" with foreseeable future nano-technology scenarios, we focus on robots with minimal capabilities. Specifically:

1. Robots are autonomous (i.e. have a separate thread of execution and control) and are equally programmed (i.e. they run the same code). Differentiation in their activities – if needed – must be established run-time on the basis of perceived morphogen gradients).
2. Robots can move through the space freely and autonomously. Although this may seem a really hard requirement at nano-scale, several works are trying to achieve this by reverse-engineering cilia- and flagella-based "motors" in bacteria (Weiss and Knight 2000).
3. Each robot is provided with a random number generator enabling a simple form of symmetry breaking and robot identification (with high probability – e.g. cast ten random numbers and let them be the robot id).
4. Each robot is provided with a short-range wireless communication mechanism enabling it to broadcast messages in its neighborhood and to receive messages sent by other robots. Apart from perceiving and propagating morphogen gradients, this also enables each robot to know how many other robots are in its neighborhood (e.g. each periodically broadcasts "I am here" messages) to estimate the local density of robots

Robots do not have other capabilities other than the ones listed before. In particular:

5. They do not perceive the location (neither direction nor distance) of other robots, they do not have any kind of long range communication mechanism, nor a global accessible data space. In other words, although a robot can perceive how many robots are in the neighborhood, it can neither perceive in which direction and at which distance a specific robot is, nor in which direction a perceived morphogen gradient decreases. To follow downhill a perceived gradient, a robot has to wander randomly until it perceives it is going in the correct direction (i.e., the morphogen gradient it was following downhill is now perceived with a decreased value).
6. They do not have any notion of time and cannot rely on any global synchronization mechanism. Differentiation of activities over time can occur only on the basis of changes in the locally perceived morphogen gradients.

From a methodological viewpoint, robots exploit morphogen gradients to self-organize their respective positions in space. In particular, starting from any spatial configuration of robots: *(i)* robots start diffusing specific types of

morphogen gradients; *(ii)* robots react to locally perceived morphogen gradients by trying to follow them downhill/uphill, or by changing their activity state possibly depending on the perceived values of the morphogen gradients; *(iii)* changes in the activity state of robots can lead to inhibiting the propagation of some morphogen gradients and/or to the diffusion of new types of morphogen gradients in the system, leading back to point *(i)*. One can then apply this process several times, with new types of morphogen gradients being propagated in different phases (and exploited in some of the ways discussed in Section 2), so as to incrementally have robots self-organize into the required shape.

Readers can have an early look at the examples in Figures 2, 3, 4 and 5, to get a clue of some shapes we have been able to obtain. Before detailing such examples, we prefer to discuss related work in the area.

Related approaches

In the last few years several approaches targeting control algorithms for pattern formation in multi-robot systems (or, which is the same, for systems of simple computational particles) have been proposed (Cao et al. 1997).

Several proposals in the area define distributed algorithms for pattern formation in robots exploiting the strong assumption that each robot, via visual observation, can determine the positions and movements of all other robots (Cao et al. 1997; Gordon et al 2003). Although this hypothesis makes it rather trivial to promote the formation of a variety of global patterns, the approach is hardly applicable to micro- or nano-robots. Issues of scalability, battery consumption, line-of-sight problems, cost of global localization, etc. all call for a strictly local perception of the environment.

Other approaches have been proposed requiring robots only local perception, but still requiring them the capability to detect the distance and the direction of neighbor robots (Bay and Unsal 1994; Fredslund and Mataric 2002). The key idea is that robots, by positioning themselves at specific distances and directions from other robots, can self-organize in a variety of regular shapes. Little is said about the possibility of making more complex shapes emerge, e.g., by making information flow from robot to robot (as in the case of morphogen gradient).

A possible alternative to promote the formation of spatial patterns, in the absence of distance and direction information, is to get inspiration from the way chemicals and crystals grow into self-organized regular structures. Approaches of this kind are explored, for instance, in (Spears and Gordon 1999; Jones and Mataric 2003; SBIR). The general idea is to exploit stateful robots capable only of sensing the internal state and the presence of other robots (either via of proximity sensing or via direct contact). Robots are deployed together in an environment and there start randomly moving. When robots keep in touch with other robots, they apply internal transition rules (based on the analysis of their own state and of that of close robots) to decide whether to “stick” to that position

or continue moving. Unfortunately, these approaches enable the direct programming of transition rules leading to very simple and regular patterns only. More complex patterns require very complex search heuristics to determine the appropriate set of transition rules leading to the desired global pattern. In any case, the process leads to the formation of static non-adaptive patterns.

Algorithms for the control of shape and motion in a modular robot (i.e. a collection of simple autonomous actuator with few degrees of freedom connected with each other) have been recently proposed (Bojinov et al. 2000; Shen et al. 2002). There, "hormones" (similar to morphogen gradients) are created and propagated through the modules of the robot. Robots' modules decide how to bend their actuators depending on the locally perceived hormones. The result is to have the robot modules self-organize into globally coherent shapes or into globally coherent motion patterns. Still, this approach (and the hormones being propagated) is strictly coupled with the hardware characteristics of the robots and of its actuators, thus missing in generality.

The approach proposed in the Amorphous Computing project (Nagpal 2002) for the spatial differentiation of the activities in an amorphous network of simple computational particles is the one that most directly relates to our work. Simple particles, by communicating only with their local neighborhood, can self-organize into various patterns of activity by propagating morphogen gradients and by changing their internal state according to the perceived morphogen gradients. The main difference from our work is that in the Amorphous Computing approach particles – unlike our robots – cannot move altering their respective topologies. Furthermore, some particles must be in a special initial state, thus requiring an a priori differentiation among particles that we do not require for our robots.

A possible objection

Algorithms to create a shared coordinate systems in a network on the basis of mere network connectivity (i.e., requiring the only capability of locally sending/receiving messages) have been recently proposed (Nagpal et al. 2003; Shang et al. 2003). Since these algorithms could be executed also by robots with minimal capabilities, they have been proposed as a way to direct robots towards specific positions in the coordinate systems, so as to make any desired pattern emerge.

Proposals in that directions (Kondacs 2003; Gordon et al. 2003; Stoy and Nagpal 2004) work roughly as follow: *(i)* each robot is a priori provided with some kind of geometrical description of the shape to form (e.g., an $f(x,y)$ function for 2D shapes); *(ii)* a shared coordinate system is set up in the system, typically be electing a particle as the origin of the system; *(iii)* Robots start moving with the goal of filling the specified shape. These movements can be

driven by some a priori heuristics or by morphogen gradients emitted by those robots that recognize that there are still portions of the shape to fill.

Although the above approach theoretically allows the building of any kind of shape with great accuracy, we do not commit to it for several reasons. On the one hand, coding a complex shape in some digital representation can be very hard. Even if effective solutions have been proposed to this purpose (Stoy and Nagpal 2004), one should also consider that such representation must be locally stored by each robot, which may not be possibly for robot with very limited resources. On the other hand, relying on an a priori representation of the shape makes the resulting formation of robots not robust and not adaptable. Robots deployed in a constrained environment where the shape does not fit would be left with no choices but leaving the shape incomplete, rather than flexibly adapt the shape. Finally, one should add that preserving a shared coordinate system in the presence of mobile nodes is feasible only if the nodes' are strictly constrained as, e.g., in the case of particles in direct contact with each other and moving accordingly to few degrees of freedom.

Based on the above consideration, we stick to our more general approach, even if this may imply more efforts to achieve specific shapes.

EXPERIMENTS

Here we describe a set of experiments discussing a variety of patterns we have been able to achieve with the morphogen gradients approach. All the results have been tested on a simulator for large-scale mobile ad-hoc networks developed within our research group (Mamei and Zambonelli 2004a). In the experiments, we assumed that robots are placed on a 2D surface, and that have no physical size or (which is the same) that they can pass through each other. Simple motion mechanisms can always be conceived for robots to "walk" around each other.

Barycenter

In this example, starting from a random distribution of robots, a sort of distributed leader election algorithm is executed to identify the robot closest to the barycenter, i.e., the "center of gravity", of the whole system. Specifically, given *n* robots in space, the barycenter is that robot that minimizes the sum of the distances to all the *n* robots.

Detecting the barycenter of the system is very important for pattern formation, in that it identifies a reasonable point from which to start subsequent shape formation activities. Morphogen gradients, expressing the approximate distance from their respective sources, enable the definition of a simple and intuitive algorithm for the identification of the barycenter.

The algorithm: Each and every robot propagates a BARYCENTER gradient whose value increases by one at each hop. Each robot senses BARYCENTER gradients propagated by all the other robots as they arrive, and adds their values together, call the resulting value *totGradients*. *totGradients* is the sum of distances to all the other robots. Therefore, the robot having the minimum *totGradients* is the barycenter. Since *totGradients* decreases monotonically to the barycenter, each robot can understand whether it has *totGradients* minimum or not, by simply comparing its value with the neighbors' ones. If no neighbors have a lower value of *totGradients*, the robot is the barycenter.

The commented pseudo code of the algorithm is in Figure 2. We emphasize the algorithm does not have a well-defined termination point. Simply, each robot keeps on waiting for the income of new BARYCENTER gradients, to evaluate over and over whether it is the barycenter of not. Eventually, the algorithm converges, and robots will no longer receive any new gradient. The evolution of a sample simulation of the barycenter election algorithms is reported in Figure 3. In this figure (as well as in following simulation figures) it can be noted that, during the process, some robots may temporarily recognize themselves as barycenter. However, at the end, a single barycenter remains.

Slight modifications of the algorithm can be defined to elect two robots, aligned around the barycenter at a specific distance from each other, as well as to identify robots on the border of the structure.

```
barycenter = false // I am not the barycenter
totGradients = 0 // sum of values received
// propagate the BARYCENTER gradient
injectGradient(BARYCENTER, uniqueNumber)
// infinite cycle
while (1)
// blocks waiting for incoming gradients
Gradients[] = getGradients(BARYCENTER);
// sum the value of all gradients
forall Gradients
totGradients+=Gradients[i].value;
  end forall
// analyse totGradients value of neighbors
count = 0 // a simple counter
forall neighbor robots
if neighbor[i].totGradients > totGradients
   count++
end if
end forall
// if all neighbors have a greater value
if numNeighbors = count
  barycenter = true // I am the barycenter
  else
  // otherwise I am not the barycenter
  barycenter = false
  end if
end while
```

FIGURE 2. Pseudo-code of the barycenter algorithm.

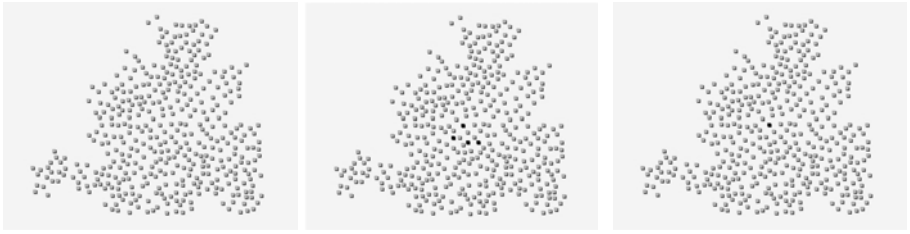

FIGURE 3. From left to right, different stages of the establishment of the barycenter in a cloud of randomly distributed robots. As the system evolves, some robots (in black) may temporarily consider themselves the barycenter. Eventually, a single barycenter is left.

Circle

In this example, robots run a distributed algorithm to cooperatively assume a circle shape.

The Algorithm: Each robot runs the barycenter algorithm described in the previous section. The resulting barycenter robot will serve as the circle center. This robot propagates a CIRCLE gradient which increases its value by one at each hop. All the other robots sense the gradient. If they sense a value greater than R (intended circle radius) they move following downhill the CIRCLE gradient. Eventually, all the robots outside the intended circle radius will collapse toward it.

```
if robot == BARYCENTER
        injectGradient(CIRCLE)
end if
while(1)
  if getGradient(CIRCLE).value > R
        moveDownhill(CIRCLE)
    end if
end while
```

FIGURE 4. Pseudo-code of the circle algorithm

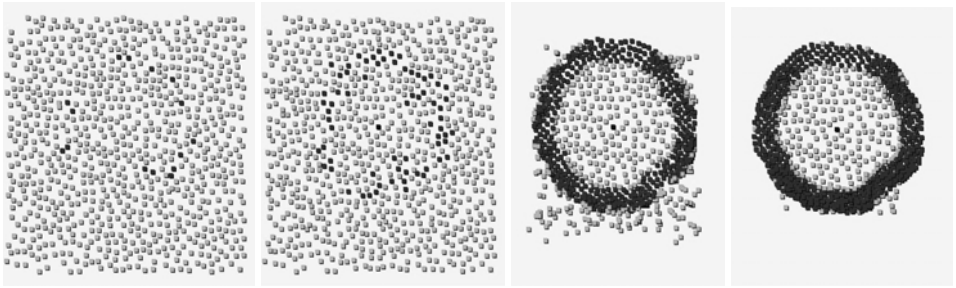

FIGURE 5. From left to right. Different stages of the circle formation. As the barycenter start propagating the circle gradient, some robots (in black) recognize themselves as being at the correct distance from the center and do not move; the other robots gradually collapse toward the circle circumference.

The pseudo code of the circle algorithm is in Figure 4. Also in this case, the algorithm does not end: simply, the robots that found themselves inside the circle will stop. The result of a sample simulation is in Figure 5.

It is important to note that, despite the fact that during the execution of the barycenter algorithm some robots may temporarily consider themselves the barycenter, this causes not harm to the circle algorithm. Simply, these robots will temporarily diffuse a spurious CIRCLE gradient.

Another remark relates to the fact that, as stated in Section 3.1, our robots cannot sense in which directions a gradient decreases. Therefore, a robot has to randomly choose a direction to move and, if the robot senses that the gradient of interest does not decrease – wrong guess – it can simple invert its direction. The key drawback of this technique (in addition to the introduction of a small overhead, as discussed in Section 5) is that it makes it possible for some robots to get lost, i.e., get disconnected from the network without any further information about where the rest of the robots are. However, since these unlucky events are extremely rare, and since individual robots are disposable, this causes no harm.

As an additional note, we emphasize that the execution of the circle algorithm, in the presence of two barycenter, enables the forming of elliptic shapes.

Ring

In this example, robots run a distributed algorithm, simply extending the circle one, to cooperatively assume a ring shape.

The Algorithm. Once the barycenter algorithm has run, and the circle has been formed (better: once robots on the circumference of the forming circle recognize their being in there), the robots on the circumference start propagating a RING gradient which increases its value by one at each hop. This RING gradient, which also propagates to the inner part of the circle, attracts robots towards the circumference, thus emptying the inside of the ring. The thickness of the ring can be tuned by having robots stop following the RING gradient when it reaches a value of T, where T will consequently be the thickness of the ring.

The pseudo code of the ring algorithm is in Figure 6. The result of a sample simulation is in Figure 7.

As for the case of the circle, we outline that by executing the ring algorithm in the presence of two barycenters “8”-like shapes can be obtained.

```
if robot == CENTER
```

```
        injectGradient(CIRCLE)
  end if
if getGradient(CIRCLE).value == R
        injectGradient(RING)
end if
if getGradient(CIRCLE).value != R
   while getGradient(RING).value >= T
         moveDownhill(RING)
   end while
end if
```

FIGURE 6. Pseudo-code of the ring algorithm.

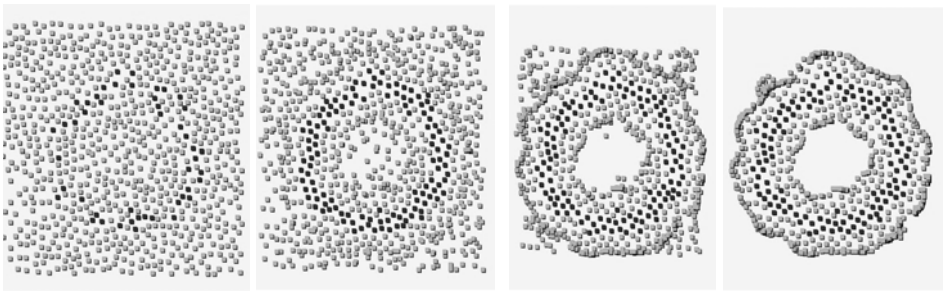

FIGURE 7. From left to right. Different stages of the ring formation. As the robots at the correct distance from the barycenter self-recognize their being there, they start injecting a gradient that attracts toward them inner robots.

Making Lobes

In this experiment, we tried to break the circular symmetry of previous experiments and let lobes emerge in the global shape. The overall idea is to exploit robots density as the source to break the symmetry. For instance, when forming a circle, robots start to collapse toward the circle itself. If the number of robots compared to the circle size is very high, then the perimeter of the circle will be very crowded. Our idea is to force robots in very crowded areas to rearrange their positions so as to stay more separate form each other (remember that our robots can sense how many other robots are in the neighborhood). This process ends up in a slight deformation of the circle (i.e., in the emergence of small "lobes") in those part of its circumference where an excess of robots are accumulating. These small emergent lobes can be amplified via an additional mechanism of morphogen gradient inhibition that, in turn, makes larger lobes emerge.

To this end, and with reference to the circle, it is worth noting that the emergence of the circle shape directly derives from having the CIRCLE gradient spread in every direction uniformly. In this way, all the robots sensing a value R of the CIRCLE gradient end up in being almost equidistant form the center or,

when the density is taken into account, in the circumference of a circle with small lobes. However, if one makes the CIRCLE gradient increase its value slower in zones of high density then, in these zones, the gradient would reach the value R farther from the source. Robots, following that gradient, would not dispose on a circle, but on a circle with a lobe, where the lobe would correspond to the place in which the gradient reaches value R farther. This leads to the following simple algorithm.

The Algorithm. Robots run the CIRCLE algorithm and, upon receiving the CIRCLE gradient, have to re-propagate it. However, before doing that, robots sense the number of other robots in their neighborhood. If the number of robots in the neighborhood exceeds a specified threshold (*criticalDensity1*), the robot sets the rate at which the gradient increases to 0. This increasing rate will be reset to the default value of 1 when the density falls below another specified threshold (*criticalDensity2*).

The pseudo code of this algorithm is in Figure 8. The result of a simple experiment is in figure 9. It is worth remarking that the algorithm does not enable to predict where lobes will form in the circle, and how many lobes will eventually form. This is an emergent characteristic of the systems that critically depends on two non controllable factors: the initial disposition of robots and the outcome of the random movement of robots towards the center. These two factors will affect the way robots will accumulate around the circle and, thus, the density of robots.

```
  if robot == CENTER
       injectGradient(CIRCLE)
end if
if getGradient(CIRCLE).value > R
       moveDownhill(CIRCLE)
else if getGradient(CIRCLE).value == R
/* making robots escape from crowd make
   small lobes emerge */
       moveAwayFromCrowd()
     end if
end if
// the following, makes even large lobes emerge
if numNeighbors > criticalDensity1
     // set the gradient increasing rate to 0
       setGradientAddValue(CIRCLE, 0)
end if
if numNeighbors < criticalDensity2
     // restore default increasing rate
       setGradientAddValue(CIRCLE, 1)
end if
```

FIGURE 8. Pseudo-code of the lobes algorithm

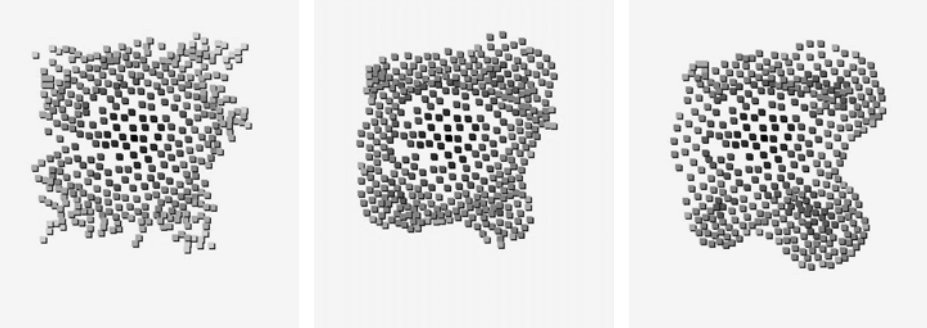

FIGURE 9. From left to right, different stages of the circle with lobes formation. The robots in the circle that detect a high density of robots, inhibit the propagation of the circle gradient, thus leading to the formation of lobes.

Polygons

The emergent phenomenon of lobes in the previous section is interesting. Still, it would be important to have a way of controlling such emergent behavior. In a further set of experiments, we tried to enrich the algorithm to control the number of lobes to be created and obtain regular polygon shapes.

```
if robot == CENTER
   injectGradient(CIRCLE)
end if
if getGradient(CIRCLE).value == R
   while not hasGradient(ELECT)
     if getGradient(ELECT).value > L
       iAmLeader();
       injectGradient(ELECT)
       setGradientAddValue(CIRCLE, 0)
     end if
     else if nextRandom() > T
       iAmLeader();
       injectGradient(ELECT)
        setGradientAddValue(CIRCLE,0)
     end if
   end while
end if
else if getGradient(CIRCLE).value > R
        if getGradient(ELECT).value == 1
          setGradientAddValue(CIRCLE, 0)
        end if
       end if
end if
while(1)
 if getGradient(CIRCLE).value > R
   moveDownhill(CIRCLE)
 else if getGradient(CIRCLE).value == R
           moveAwatFromCrowd ()
 end if
end while
```

FIGURE 10. Pseudo-code of the polygons algorithm.

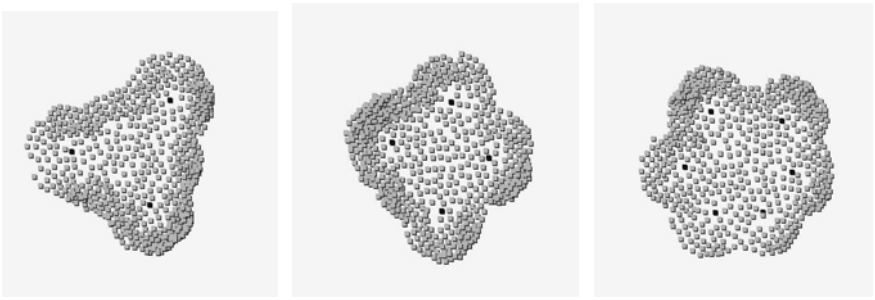

FIGURE 11. Different polygon shapes obtained by multiple lobes.

The idea to control the number of lobes is again rooted on a leader election mechanism. We want to design an algorithm to elect *n* leaders on the circumference of the circle. These leaders must be equidistant from one another. Once this has been accomplished, the leaders will be in charge of adopting the mechanism described in the previous section, i.e., not to increase the value of the CIRCLE gradient being re-propagated, so as to force the emergence of *n* lobes equidistant from one another and, consequently, leading to a nearly regular *n*-polygon shape.

The Algorithm. (i) each node runs the circle algorithm, *(ii)* once a robot on the circumference of the forming circle recognize its being in the current position, it start casting random numbers; *(iii)* each node casting a number greater than a specified threshold becomes a leader – the threshold (T) is chosen so that it is very unlikely that two nodes becomes leaders shortly one after another; *(iv)* the leader starts propagating an ELECT gradient, that propagates only in the circle perimeter region (i.e., robots that are not on the RING inhibits its propagation); *(v)* nodes receiving the ELECT gradient, stop casting random numbers and if the received gradient value overcomes another specified threshold *L,* they become leaders on their turn; *(vi)* each leader sets the ELECT gradient value to 0 and continues its propagation; *(vii)* once the ELECT gradient is fully propagated there should be almost *(circle-length)/L* equidistant leaders on the circle. Thus *L* is a parameter controlling which polygon will emerge.

The pseudo code of the algorithm is in Figure 10. The results of some experiments, showing that the algorithm enables to control the emergence of regular polygon shapes are in Figure 11.

PERFORMANCE EVALUATION

Validating our approach in terms of performances basically amounts at verifying *(i)* that it is reasonably scalable and *(ii)* that the assumption of minimal capabilities is not too much penalizing. The results are presented in Figure 12 and refer to a virtual time that adopts as the unity '1' the time taken to propagate a gradient between to neighbor robots. Referring to an actual time requires assumptions on hardware and motion speed that are not relevant here.

With regard to point *(i)*, we have verified that our approach scales linearly with the number of robots: the time for a swarm of randomly placed robots to reach a stable configuration increases linearly with the number of robots. With regard to point *(ii)* we have limited our attention at verifying that the impact of the assumption of non-directional sensing is not too penalizing. We have compared the time required by robots to self-organize into specific shapes with and without the capability to perceive the direction to which a morphogen gradient is decreasing. The result is that the overhead caused by robots wandering randomly to properly detect in which direction to go is very limited, independently of the specific shape to be obtained and independently of the overall number of robots in the system.

Comparing our approach with other approaches based on robots with much powerful capabilities (i.e., global sensing and a priori knowledge of the geometrical shape to obtain) is simply meaningless. In fact, the notably better performances that these approaches would obviously exhibit would be obtained at the price of notably increasing robots' complexity, size and price.

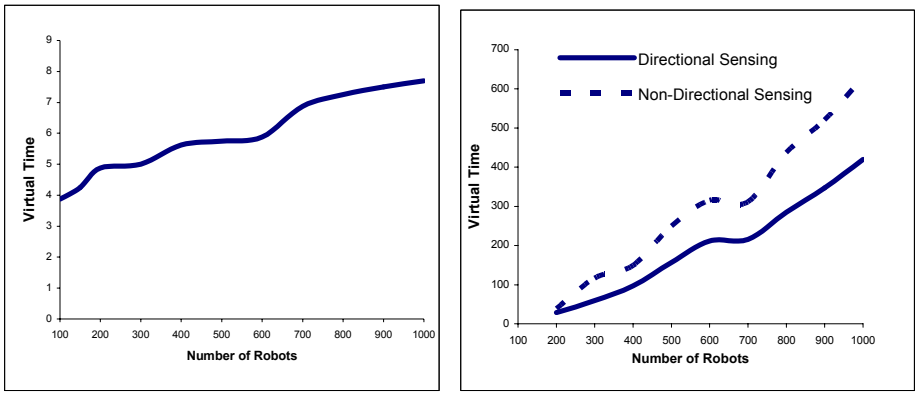

FIGURE 12. Performance evaluation. (left) Identification of the barycenter. The time required to elect a barycenter grows linearly with the number of robots. (right) Ring formation. The time required for the shaping of robots into a ring grows linearly with the number of robots. Furthermore, the assumption of non-directional sensing is not highly penalizing, if compared with the results achieved by robots capable of directional sensing.

CONCLUSIONS AND OPEN DIRECTIONS

The morphogen gradient approach enables the effective formation of a variety of complex shapes in computational robots with minimal capabilities.

Despite these promising results, several open directions are still to be investigated. Those of interest to our research group include: experiencing differentiation of activities and global coordination based on cellular automata inspired approach; defining a simple and modular programming model, enabling designer and programmers to enforce in a modular and compositional way a variety of complex patterns; achieving – other than the formation of static patterns – coherent motion gaits in robots; building some hardware prototype for robots and validating our approach in the real world.

REFERENCES

Bay, J. S., and C. Unsal. 1994. Spatial Self-Organization in Large Populations of Mobile Robots. *IEEE International Symposium on Intelligent Control,* Columbus (OH).

Bojinov, H., A. Casal, and T. Hogg. 2000. Emergent Structures in Modular Self-Reconfigurable Robots. *IEEE Intl. Conf. on Robotics and Automation,* San Francisco (CA).

Bonabeau, E. 1997. From Classical Models of Morphogenesis to Agent-based Models of Pattern Formation, *Artificial Life*, 3:191-211.

Cao, Y. U., A. S. Fukunaga, A. B. Kahng, and F. Meng. 1997. Cooperative Mobile Robotics: Antecedents and Directions. *IEEE Int. Conf. on Intelligent Robots and Systems*, Yokoama (J).

Coore, D. 1999. *Botanical Computing: A Developmental Approach to Generating Interconnect Topologies on an Amorphous Computer*. PhD Thesis, MIT, Boston (MA).

Day, S. J., and P. A. Lawrence. 2000. Morphogens: Measuring Dimensions: the Regulation of Size and Shape. *Development* 127:2977-2987.

Estrin, D., D. Culler, K. Pister, and G. Sukjatme. 2002. Connecting the Physical World with Pervasive Networks. *IEEE Pervasive Computing*, 1(1):59-69.

Fredslund, J., and M. J. Mataric. 2002. A General Algorithm for Robot Formations Using Local Sensing and Minimal Communication. *IEEE Transactions on Robotics and Automation*, 18(5):837-846.

Gordon, N., I. A. Wagner, and A. M. Bruckstein. 2003. Discrete Bee Dance Algorithm for Pattern Formation on a Grid. *IEEE International Conference on Intelligents Agents Technologies*, Toronto (CA).

Guo, Y., G. Poulton, P. Valencia, and G. James. 2004. Designing Self-Assembly for 2-Dimensional Building Blocks. *Engineering Self-Organizing Applications*, LNCS No. 1977, Springer Verlag.

Jones, C., and M. J. Mataric. 2003. From Local to Global Behavior in Intelligent Self-Assembly. *Submitted to the IEEE Conference on Robotics and Automation*.

Kondacs, A., 2003. Biologically-inspired Self-Assembly of 2D Shapes Using Global-to-local Compilation. *International Joint Conference on Artificial Intelligence (IJCAI)*, Acapulco (MX).

Lawrence, P.A. 1992. *The Making of a Fly: the Genetics of Animal Design*. Blackwell Science, London (UK).

Mamei, M., and F. Zambonelli. 2004a. Programming Mobile and Pervasive Computing Applications with the TOTA Middleware. *IEEE Conference on Pervasive Computing and Communications*, Orlando (FL).

Mamei, M., and F. Zambonelli. 2004b. Co-Fields: a Physically Inspired Approach to Distributed Motion Coordination. *IEEE Pervasive Computing*, 3(2):52-60.

Nagpal, R. 2002. Programmable Self-Assembly Using Biologically-Inspired Multirobot Control. *ACM Joint Conference on Autonomous Agents and Multiagent Systems*, Bologna (I).

Nagpal, R., H. Shrobe, and J. Bachrach. 2003. Organizing a Global Coordinate System from Local Information on an Ad Hoc Sensor Network. *Information Processing in Sensor Networks*, LNCS No. 2643, Springer Verlag, Berlin (D).

Poor, R. 2001. *Embedded Networks: Pervasive, Low-Power, Wireless Connectivity*. PhD Thesis, MIT, Boston (MA).

Pister, K. 2003. Invited Plenary Talk. *The 23rd International Conference on Distributed Computing Systems*, Providence (RI).

SBIR Project. A Cooperative Multi-Robot Control Architecture. Tech. Rep. No. 01168, Dynamic Concepts Inc.

Shang, M., W. Ruml, and Y. Zhang. 2003. Localization from Mere Connectivity, *ACM Conference on Mobile ad-hoc Computing and Networking*, Annapolis (MD).

Shen, W. M., B. Salemi, and P. Will. 2002. Hormone-Inspired Adaptive Communication and Distributed Control for CONRO Self-Reconfigurable Robots. *IEEE Transactions on Robotics and Automation*, 18(5):1-12.

Spears, W. M., and D.F. Gordon. 1999. Using Artificial Physics to Control Robots. *IEEE International Conference on Information, Intelligence and Systems*.

Stoy, K., Nagpal R. 2004. Self-Reconfiguration Using Directed Growth, *7th International Symposium on Distributed Autonomous Robotic Systems*, Toulouse (F).

Weiss, R., Knight T. 2000. Engineered Communications for Microbial Robotics, *DNA6 Sixth International Meeting on DNA Based Computers*, Leiden (NE).

Wolpert, L. 1998. *Principles of Development*. Oxford University Press, Oxford (UK).